Rééchantillonnage gaussien en grande dimension pour les problèmes inverses

Clément Gilavert, Saïd Moussaoui, Jérôme Idier

Rééchantillonnage gaussien en grande dimension pour les problèmes inverses

Clément GILAVERT, Saïd MOUSSAOUI, Jérôme IDIER

IRCCYN, CNRS UMR 6597, 1, rue de la Noë, BP 92101, F-44321 Nantes Cedex 03, France
prenom.nom@irccyn.ec-nantes.fr

Résumé – Nous nous intéressons aux algorithmes MCMC dans le cas où une des composantes à rééchantillonner est un vecteur gaussien de grande dimension dont la covariance varie au cours des itérations (du fait de sa dépendance à des hyperparamètres eux-même rééchantillonnés par exemple). Pour éviter le coût de calcul d'une factorisation de type Cholesky à chaque itération, une solution récente propose de passer par la résolution d'un système linéaire de grande dimension, effectuée de façon approchée par gradient conjugué tronqué. Notre contribution consiste (1) à mettre en évidence que cette troncature empêche la convergence vers la loi cible ; (2) à proposer l'incorporation d'une étape d'acceptation-rejet peu coûteuse rétablissant la convergence. Nous illustrons nos résultats sur un problème de super-résolution en imagerie. Ils mettent en avant de façon concrète le biais engendré par la version sans étape d'acceptation-rejet, ainsi que le degré de troncature possible grâce à l'algorithme proposé.

Abstract – The resolution of large-scale linear inverse problems using MCMC methods may require a step of drawing samples from a high dimensional Gaussian distribution which covariance matrix changes along iterations (for example when it depends on hyperparameters that need to be sampled as well). Direct Gaussian sampling methods like Cholesky factorization may induce an excessive numerical complexity but a recent work sugguested to solve a linear system to obtain a new sample, with a truncated conjugate gradient. Our contribution consists in (1) highlighting that this truncation prevents the convergence to the target distribution ; (2) proposing to add an accept-reject step to restore the convergence. We illustrate our method with an unsupervised super-resolution problem in image processing. Results show clearly the bias induced without the accept-reject step and the possible truncation level with the proposed algorithm.

1 Introduction

En inférence bayésienne, il est fréquent qu'une partie des composantes à rééchantillonner correspondent à un vecteur ou à une image de grande dimension et de loi conditionnelle gaussienne. C'est souvent le cas par exemple en restauration et en reconstruction de signaux et d'images. Dans la suite, nous considérons en particulier une structure de problème inverse linéaire donné par le modèle d'observation $\boldsymbol{y} = \boldsymbol{M}\boldsymbol{x} + \boldsymbol{b}$ avec $\boldsymbol{y}$ la mesure, $\boldsymbol{M}$ la matrice d'observation, $\boldsymbol{b}$ le bruit additif et $\boldsymbol{x} \in \mathbb{R}^N$ le vecteur d'intérêt. Supposons des modèles gaussiens $\mathcal{N}(\boldsymbol{\mu}_b, \boldsymbol{R}_b)$ et $\mathcal{N}(\boldsymbol{\mu}_x, \boldsymbol{R}_x)$ pour le bruit et la variable d'intérêt. Notons $\boldsymbol{\Theta}$ l'ensemble des hyperparamètres dont dépendent $\{\boldsymbol{\mu}_b, \boldsymbol{R}_b, \boldsymbol{\mu}_x, \boldsymbol{R}_x\}$. Le problème d'inférence bayésienne considéré ici consiste à estimer à la fois $\boldsymbol{x}$ et $\boldsymbol{\Theta}$. Pour cela une approche de type MCMC est principalement utilisée pour tirer des échantillons de la loi *a posteriori* $P(\boldsymbol{x}, \boldsymbol{\Theta}|\boldsymbol{y})$ afin d'en extraire des estimateurs. Une structure de Gibbs par bloc est souvent utilisée pour le rééchantillonnage. A chaque itération k :

1. Tirer $\boldsymbol{\Theta}^{(k)}$ suivant la loi conditionnelle $P(\boldsymbol{\Theta}|\boldsymbol{x}^{(k-1)}, \boldsymbol{y})$.
2. Tirer $\boldsymbol{x}^{(k)}$ suivant la loi conditionnelle $P(\boldsymbol{x}|\boldsymbol{\Theta}^{(k)}, \boldsymbol{y})$.

On montre facilement que la loi conditionnelle $P(\boldsymbol{x}|\boldsymbol{\Theta}, \boldsymbol{y})$ est une loi gaussienne dont la matrice de précision (inverse de la matrice de covariance) et la moyenne sont données par

$$\boldsymbol{Q} = \boldsymbol{M}^{\mathrm{t}}\boldsymbol{R}_b^{-1}\boldsymbol{M} + \boldsymbol{R}_x^{-1} \tag{1}$$

$$\boldsymbol{Q}\boldsymbol{\mu} = \boldsymbol{M}^{\mathrm{t}}\boldsymbol{R}_b^{-1}(\boldsymbol{y} - \boldsymbol{\mu}_b) + \boldsymbol{R}_x^{-1}\boldsymbol{\mu}_x \tag{2}$$

En grande dimension, deux difficultés apparaissent dans l'étape 2 du rééchantillonnage : la résolution du système pour calculer $\boldsymbol{\mu}$, puis l'échantillonnage de la gaussienne de matrice de précision $\boldsymbol{Q}$.

L'approche la plus classique consiste à inverser la matrice de précision $\boldsymbol{R} = \boldsymbol{Q}^{-1}$, à calculer sa factorisation de Cholesky $\boldsymbol{R} = \boldsymbol{L}_r\boldsymbol{L}_r^{\mathrm{t}}$, puis à l'utiliser pour générer le nouvel échantillon $\boldsymbol{x} = \boldsymbol{R}\boldsymbol{Q}\boldsymbol{\mu} + \boldsymbol{L}_r\boldsymbol{\omega}$, avec $\boldsymbol{\omega} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. Une alternative [1] propose d'effectuer la factorisation de Cholesky directement sur la matrice de précision $\boldsymbol{Q} = \boldsymbol{L}_q\boldsymbol{L}_q^{\mathrm{t}}$ puis de calculer le nouvel échantillon

$$\boldsymbol{x} = \boldsymbol{L}_q^{-\mathrm{t}}\left(\boldsymbol{L}_q^{-1}\boldsymbol{Q}\boldsymbol{\mu} + \boldsymbol{\omega}\right) \tag{3}$$

avec $\boldsymbol{\omega} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, ce qui revient à résoudre deux systèmes triangulaires.

Mais ces approches nécessitent $O(N^3)$ opérations si la matrice n'a pas de structure spécifique, ce qui les rend infaisables en grande dimension. Néanmoins si la matrice $\boldsymbol{Q}$ est creuse, calculer sa factorisation de Cholesky devient possible [1, 2]. Par ailleurs, si $\boldsymbol{Q}$ est circulante [3], effectuer les calculs dans le domaine fréquentiel à l'aide de transformées de Fourier permet de réduire fortement la complexité.

Une autre approche appelée "Perturbation Optimization" [4, 5, 6] consiste à simuler un échantillon $\boldsymbol{\eta} \sim \mathcal{N}(\boldsymbol{Q\mu}, \boldsymbol{Q})$, puis à résoudre le système $\boldsymbol{Qx} = \boldsymbol{\eta}$. Il est facile de vérifier que la loi de $\boldsymbol{x}$ est alors bien $\mathcal{N}\left(\boldsymbol{\mu}, \boldsymbol{Q}^{-1}\right)$. Dans le cadre étudié ici, le tirage de $\boldsymbol{\eta}$ peut se faire rapidement de la façon suivante :

1. Tirer $\boldsymbol{\eta}_b \sim \mathcal{N}(\boldsymbol{y} - \boldsymbol{\mu}_b, \boldsymbol{R}_b)$.
2. Tirer $\boldsymbol{\eta}_x \sim \mathcal{N}(\boldsymbol{\mu}_x, \boldsymbol{R}_x)$.
3. Poser $\boldsymbol{\eta} = \boldsymbol{M}^{\mathrm{t}} \boldsymbol{R}_b^{-1} \boldsymbol{\eta}_b + \boldsymbol{R}_x^{-1} \boldsymbol{\eta}_x$.

Notons que le tirage de $\boldsymbol{\eta}$ est peu coûteux si $\boldsymbol{R}_b$ et $\boldsymbol{R}_x$ ont des structures simples. En revanche le coût de l'inversion du système linéaire $\boldsymbol{Qx} = \boldsymbol{\eta}$ reste $O(N^3)$ opérations. C'est pourquoi plusieurs travaux [4, 5, 6] préconisent de résoudre ce système par gradient conjugué en effectuant un nombre réduit $K \ll N$ d'itérations. Cependant, l'effet de cette troncature en termes de convergence de la chaîne de Markov n'a pas été étudié pour l'instant.

Notre principale contribution est une modification de l'algorithme de *Perturbation Optimization* : nous proposons d'utiliser cet algorithme pour proposer un candidat, puis d'accepter cette proposition selon une étape d'acceptation-rejet dont le ratio est calculé d'après la théorie des sauts réversibles [7]. Cette étape permet de garantir la convergence vers la loi cible, quel que soit le degré de troncature dans la résolution du système. De plus, le calcul du taux d'acceptation étant peu coûteux, celui-ci peut être utilisé pour contrôler le niveau de troncature, c'est-à-dire arrêter la résolution du système quand le taux d'acceptation atteint un seuil prédéfini.

La méthode proposée est illustrée sur un problème de super-résolution en imagerie. Les résultats mettent en avant le biais engendré par la version sans étape d'acceptation-rejet, ainsi que le degré de troncature possible grâce à l'algorithme proposé.

2 Echantillonner une loi gaussienne à l'aide des MCMC à sauts réversibles

2.1 Cadre général

Afin de définir un schéma convergent, nous nous plaçons dans le cadre des MCMC à sauts réversibles [7]. Bien qu'ils soient principalement utilisés pour échantillonner des lois de dimension variable, nous nous plaçons ici dans le cas où la dimension reste constante. Cette approche s'appuie sur la simulation d'une variable auxiliaire $\boldsymbol{z} \sim P_{\boldsymbol{Z}}(\boldsymbol{z}|\boldsymbol{x}^-)$, où $\boldsymbol{x}^-$ désigne l'échantillon précédent, et l'application d'une transformation du couple $(\boldsymbol{x}^-, \boldsymbol{z})$, pour produire un couple de candidats $(\boldsymbol{x}, \boldsymbol{s}) = \boldsymbol{\phi}(\boldsymbol{x}^-, \boldsymbol{z})$ [1]. Cette transformation déterministe doit également être réversible, c'est-à-dire $\boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{s}) = (\boldsymbol{x}^-, \boldsymbol{z})$. Ainsi, le candidat $\boldsymbol{x}$ issu de la transformation est soumis à une règle d'acceptation-rejet, pour définir le nouvel échantillon $\boldsymbol{x}^+$, avec un taux d'acceptation donné par :

$$\alpha = \min\left(1, \frac{P_{\boldsymbol{X}}(\boldsymbol{x}) P_{\boldsymbol{Z}}(\boldsymbol{s}|\boldsymbol{x})}{P_{\boldsymbol{X}}(\boldsymbol{x}^-) P_{\boldsymbol{Z}}(\boldsymbol{z}|\boldsymbol{x}^-)} |\Phi|\right) \tag{4}$$

1. Notons que dans notre cas, la variable $\boldsymbol{s}$ ne sera pas utilisée.

avec Φ le jacobien de la transformation $\boldsymbol{\phi}$. Ce cadre général des MCMC à sauts réversibles laisse donc comme degrés de liberté le choix de la distribution $P_{\boldsymbol{Z}}$ de la variable auxiliaire ainsi que la transformation déterministe $\boldsymbol{\phi}$.

Pour l'échantillonnage d'un vecteur gaussien $\boldsymbol{x} \sim \mathcal{N}\left(\boldsymbol{\mu}, \boldsymbol{Q}^{-1}\right)$ défini comme en introduction, nous proposons une généralisation de l'approche introduite dans [8], en simulant $\boldsymbol{z}$ suivant une loi normale $P_{\boldsymbol{Z}}(\boldsymbol{z}|\boldsymbol{x}^-) = \mathcal{N}(\boldsymbol{Ax}^- + \boldsymbol{c}, \boldsymbol{B})$, et nous optons pour une transformation déterministe $\boldsymbol{\phi}$ définie par

$$\begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{s} \end{pmatrix} = \begin{pmatrix} \phi_1(\boldsymbol{x}^-, \boldsymbol{z}) \\ \phi_2(\boldsymbol{x}^-, \boldsymbol{z}) \end{pmatrix} = \begin{pmatrix} -\boldsymbol{x}^- + \boldsymbol{f}(\boldsymbol{z}) \\ \boldsymbol{z} \end{pmatrix} \tag{5}$$

Le choix des paramètres $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{c}$ et de la fonction $\boldsymbol{f}(\cdot)$ permet d'aboutir à des schémas convergents dont les performances en terme de taux d'acceptation et de correlation seront différents.

Le taux d'acceptation peut s'exprimer en fonction de ces nouveaux paramètres : d'après (5) nous avons $|\Phi| = 1$, le ratio devient donc

$$\alpha = \min\left(1, e^{-\frac{1}{2}\Delta S}\right) \tag{6}$$

avec

$$\begin{aligned} \Delta S \;=\; & (\boldsymbol{x} - \boldsymbol{\mu})^{\mathrm{t}} \boldsymbol{Q} (\boldsymbol{x} - \boldsymbol{\mu}) - (\boldsymbol{x}^- - \boldsymbol{\mu})^{\mathrm{t}} \boldsymbol{Q} (\boldsymbol{x}^- - \boldsymbol{\mu}) \\ & + (\boldsymbol{z} - \boldsymbol{Ax} - \boldsymbol{c})^{\mathrm{t}} \boldsymbol{B}^{-1} (\boldsymbol{z} - \boldsymbol{Ax} - \boldsymbol{c}) \\ & - (\boldsymbol{z} - \boldsymbol{Ax}^- - \boldsymbol{c})^{\mathrm{t}} \boldsymbol{B}^{-1} (\boldsymbol{z} - \boldsymbol{Ax}^- - \boldsymbol{c}) \end{aligned}$$

En utilisant (5) on montre que ΔS se simplifie en

$$\begin{aligned} & \Delta S = \left(\boldsymbol{x} - \boldsymbol{x}^-\right)^{\mathrm{t}} * \\ & \left[\left(\boldsymbol{Q} + \boldsymbol{A}^{\mathrm{t}} \boldsymbol{B}^{-1} \boldsymbol{A}\right) \boldsymbol{f}(\boldsymbol{z}) - 2\left(\boldsymbol{Q\mu} + \boldsymbol{A}^{\mathrm{t}} \boldsymbol{B}^{-1} (\boldsymbol{z} - \boldsymbol{c})\right)\right] \end{aligned} \tag{7}$$

2.2 Résultats préliminaires et liens avec les méthodes existantes

Résultat 1. *La probabilité d'acceptation α est toujours égale à un si*

$$\boldsymbol{f}(\boldsymbol{z}) = 2\left(\boldsymbol{Q} + \boldsymbol{A}^{\mathrm{t}} \boldsymbol{B}^{-1} \boldsymbol{A}\right)^{-1} \left(\boldsymbol{Q\mu} + \boldsymbol{A}^{\mathrm{t}} \boldsymbol{B}^{-1} (\boldsymbol{z} - \boldsymbol{c})\right) \tag{8}$$

Démonstration. Remplacer $\boldsymbol{f}(\boldsymbol{z})$ dans l'equation (7) donne $\Delta S = 0$ et donc $\alpha = 1$. □

Notons que ce premier résultat fait apparaitre la résolution exacte d'un système linéaire.

Résultat 2. *Si $\boldsymbol{f}(\cdot)$ est définie par* (8)*, alors la correlation entre deux échantillons successifs est nulle si et seulement si $\boldsymbol{A}^{\mathrm{t}} \boldsymbol{B}^{-1} \boldsymbol{A} = \boldsymbol{Q}$.*

Démonstration. D'après (8), on a

$$\begin{aligned} \boldsymbol{x} = -\boldsymbol{x}^- + 2\left(\boldsymbol{Q} + \boldsymbol{A}^{\mathrm{t}} \boldsymbol{B}^{-1} \boldsymbol{A}\right)^{-1} (\boldsymbol{Q\mu} \\ + \boldsymbol{A}^{\mathrm{t}} \boldsymbol{B}^{-1} (\boldsymbol{z} - \boldsymbol{c})\big). \end{aligned}$$

De plus $\boldsymbol{z}$ est un échantillon de $\mathcal{N}(\boldsymbol{Ax}^- + \boldsymbol{c}, \boldsymbol{B})$, on peut donc écrire $\boldsymbol{z} = \boldsymbol{Ax}^- + \boldsymbol{c} + \boldsymbol{\omega}_B$ avec $\boldsymbol{\omega}_B$ totalement indépendant de $\boldsymbol{x}^-$. On obtient

$$\begin{aligned} \boldsymbol{x} = -\boldsymbol{x}^- + 2\left(\boldsymbol{Q} + \boldsymbol{A}^{\mathrm{t}} \boldsymbol{B}^{-1} \boldsymbol{A}\right)^{-1} (\boldsymbol{Q\mu} \\ + \boldsymbol{A}^{\mathrm{t}} \boldsymbol{B}^{-1} \boldsymbol{Ax}^- + \boldsymbol{A}^{\mathrm{t}} \boldsymbol{B}^{-1} \boldsymbol{\omega}_B\big) \end{aligned}$$

Enfin, le résultat 1 garantit l'acceptation de tous les échantillons, on peut donc exprimer la corrélation

$$E\left\{\boldsymbol{x}^{+}\boldsymbol{x}_{-}^{\mathrm{t}}\right\} = \left(2\left(\boldsymbol{Q}+\boldsymbol{A}^{\mathrm{t}}\boldsymbol{B}^{-1}\boldsymbol{A}\right)^{-1}\boldsymbol{A}^{\mathrm{t}}\boldsymbol{B}^{-1}\boldsymbol{A}-\boldsymbol{I}\right)\boldsymbol{Q}^{-1}$$

qui est nulle si et seulement si $\boldsymbol{A}^{\mathrm{t}}\boldsymbol{B}^{-1}\boldsymbol{A}=\boldsymbol{Q}$. □

De nombreux choix peuvent respecter la condition du résultat 2. Un premier exemple consite à considérer la factorisation de Cholesky de $\boldsymbol{Q}=\boldsymbol{L}_q\boldsymbol{L}_q^{\mathrm{t}}$ et à prendre $\boldsymbol{A}=\boldsymbol{L}_q^{\mathrm{t}}$, $\boldsymbol{B}=\boldsymbol{I}$ et $\boldsymbol{c}=\boldsymbol{0}$. Ob obtient $\boldsymbol{z}=\boldsymbol{L}_q^{\mathrm{t}}\boldsymbol{x}^{-}+\boldsymbol{\omega}$ avec $\boldsymbol{\omega}\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{I})$, et en définissant $\boldsymbol{f}(\cdot)$ par (8), le nouvel échantillon devient

$$\begin{aligned}\boldsymbol{x}^{+} &= -\boldsymbol{x}^{-}+\left(\boldsymbol{L}_q\boldsymbol{L}_q^{\mathrm{t}}\right)^{-1}\left(\boldsymbol{Q}\boldsymbol{\mu}+\boldsymbol{L}_q\boldsymbol{z}\right)\\ &= -\boldsymbol{x}^{-}+\boldsymbol{L}_q^{-\mathrm{t}}\left(\boldsymbol{L}_q^{-1}\boldsymbol{Q}\boldsymbol{\mu}+\boldsymbol{\omega}\right)+\left(\boldsymbol{L}_q\boldsymbol{L}_q^{\mathrm{t}}\right)^{-1}\left(\boldsymbol{L}_q\boldsymbol{L}_q^{\mathrm{t}}\right)\boldsymbol{x}^{-}\\ &= \boldsymbol{L}_q^{-\mathrm{t}}\left(\boldsymbol{L}_q^{-1}\boldsymbol{Q}\boldsymbol{\mu}+\boldsymbol{\omega}\right)\end{aligned}$$

On retrouve ici la mise à jour (3) qui correspond à la méthode proposée dans [1].

Un autre exemple consiste à prendre $\boldsymbol{A}=\boldsymbol{B}=\boldsymbol{Q}$, $\boldsymbol{c}=\boldsymbol{Q}\boldsymbol{\mu}$, ce qui conduit, d'après (8), à une fonction $\boldsymbol{f}(\boldsymbol{z})=\boldsymbol{Q}^{-1}\boldsymbol{z}$, c'est-à-dire la résolution exacte du système linéaire $\boldsymbol{Q}\boldsymbol{u}=\boldsymbol{z}$. Le schéma de simulation résultant est :

1. Tirer la variable auxiliaire $\boldsymbol{z}\sim\mathcal{N}\left(\boldsymbol{Q}\boldsymbol{x}^{-}+\boldsymbol{Q}\boldsymbol{\mu},\boldsymbol{Q}\right)$.
2. Calculer $\boldsymbol{u}^{*}=\boldsymbol{f}(\boldsymbol{z})$ par résolution du système $\boldsymbol{Q}\boldsymbol{u}=\boldsymbol{z}$.
3. Mettre à jour l'échantillon selon $\boldsymbol{x}^{+}=-\boldsymbol{x}^{-}+\boldsymbol{u}^{*}$.

Ce schéma correspond strictement à l'algorithme de *Perturbation Optimization* introduit en section 1. En effet, la variable auxiliaire peut s'écrire

$$\boldsymbol{z}=\boldsymbol{Q}\boldsymbol{x}^{-}+\boldsymbol{\eta} \tag{9}$$

avec $\boldsymbol{\eta}\sim\mathcal{N}\left(\boldsymbol{Q}\boldsymbol{\mu},\boldsymbol{Q}\right)$, dans ce cas $\boldsymbol{x}^{+}=-\boldsymbol{x}^{-}+\boldsymbol{Q}^{-1}\left(\boldsymbol{Q}\boldsymbol{x}^{-}+\boldsymbol{\eta}\right)=\boldsymbol{Q}^{-1}\boldsymbol{\eta}$. Ce schéma se résume à simuler la variable $\boldsymbol{\eta}$ et à retenir $\boldsymbol{x}^{+}=\boldsymbol{x}^{*}$, solution exacte de $\boldsymbol{Q}\boldsymbol{x}=\boldsymbol{\eta}$.

2.3 Algorithme proposé : le RJ-PO

L'algorithme proposé s'appuie sur le même choix que précédemment $\boldsymbol{A}=\boldsymbol{B}=\boldsymbol{Q}$, $\boldsymbol{c}=\boldsymbol{Q}\boldsymbol{\mu}$ mais n'utilise pas la fonction $\boldsymbol{f}(\cdot)$ telle que définie en (8). Afin de pouvoir considérer une résolution partielle du système précédent, on pose $\boldsymbol{f}(\boldsymbol{z})=\boldsymbol{Q}^{-1}(\boldsymbol{z}-\boldsymbol{r})$, $\boldsymbol{r}$ s'interprétant comme le résidu d'une résolution tronquée. Ce choix aboutit au résultat suivant :

Résultat 3. *Choisir* $\boldsymbol{A}=\boldsymbol{B}=\boldsymbol{Q}$, $\boldsymbol{c}=\boldsymbol{Q}\boldsymbol{\mu}$ *et* $\boldsymbol{f}(\boldsymbol{z})=\boldsymbol{Q}^{-1}(\boldsymbol{z}-\boldsymbol{r})$ *implique un taux d'acceptation* α *donné par*

$$\alpha=\min\left(1,e^{-\frac{1}{2}\Delta S}\right)$$

avec

$$\Delta S=2\boldsymbol{r}^{\mathrm{t}}\left(\boldsymbol{x}^{-}-\boldsymbol{x}\right). \tag{10}$$

Démonstration. Remplacer $\boldsymbol{f}(\boldsymbol{z})$ dans l'équation (7) amène à ce résultat. □

Par conséquent, ne retenir qu'une solution approchée du système n'empêche pas la convergence vers la loi cible si les propositions sont acceptées avec la probabilité α.

De plus, d'après l'équation (9), le tirage de la variable auxiliaire $\boldsymbol{z}$ se réduit au tirage de $\boldsymbol{\eta}\sim\mathcal{N}\left(\boldsymbol{Q}\boldsymbol{\mu},\boldsymbol{Q}\right)$ dont une méthode a été donnée en section 1.

Le schéma de simulation que nous proposons est donc :

- Tirer la variable auxiliaire $\boldsymbol{\eta}\sim\mathcal{N}\left(\boldsymbol{Q}\boldsymbol{\mu},\boldsymbol{Q}\right)$.
- Calculer $\boldsymbol{z}=\boldsymbol{Q}\boldsymbol{x}^{-}+\boldsymbol{\eta}$.
- Calculer $\tilde{\boldsymbol{u}}$, solution approchée du système $\boldsymbol{Q}\boldsymbol{u}=\boldsymbol{z}$.
- Calculer la proposition $\boldsymbol{x}=-\boldsymbol{x}^{-}+\tilde{\boldsymbol{u}}$.
- Calculer le taux d'acceptation $\alpha=\min(1,e^{\boldsymbol{r}^{\mathrm{t}}(\boldsymbol{x}-\boldsymbol{x}^{-})})$.
- Poser $\boldsymbol{x}^{+}=\boldsymbol{x}$ avec la probabilité α, sinon rejeter $\boldsymbol{x}$ en posant $\boldsymbol{x}^{+}=\boldsymbol{x}^{-}$.

En pratique il est difficile de connaitre à priori le nombre d'itération du gradient conjugué à effectuer pour obtenir un taux d'acceptation non nul. Classiquement un seuil sur la norme du résidu relatif est utilisé comme critère d'arrêt néanmoins ici cette approche ne semble pas appropriée. En revanche le calcul du taux d'acceptation ne consiste qu'en un simple produit scalaire, il est donc peu coûteux et peut être utilisé directement pour contrôler le degré de troncature du gradient conjugué. Ainsi la résolution du système se poursuit jusqu'à l'obtention d'une probabilité d'acceptation au moins égale à une valeur cible α_c prédéfinie.

L'algorithme 1 présente la méthode proposée : une première étape (lignes 2-12) consiste à inverser partiellement le système par gradient conjugué en utilisant directement le taux d'acceptation comme critère d'arrêt ; s'en suit (lignes 13-19) l'étape d'acceptation-rejet.

Algorithme 1 RJ-PO

Entrées : $\boldsymbol{\eta}\sim\mathcal{N}\left(\boldsymbol{Q}\boldsymbol{\mu},\boldsymbol{Q}\right),\boldsymbol{Q},\boldsymbol{x}^{-},\boldsymbol{u}_0,\alpha_c$
Sortie : $\boldsymbol{x}^{+}$

1: $\boldsymbol{z}=\boldsymbol{Q}\boldsymbol{x}^{-}+\boldsymbol{\eta}$
2: $\boldsymbol{r}_0=\boldsymbol{z}-\boldsymbol{Q}\boldsymbol{u}_0$
3: $\boldsymbol{d}_0=\boldsymbol{r}_0$
4: $j=0$
5: **tant que** $\boldsymbol{r}_j^{\mathrm{t}}(\boldsymbol{u}_j-2\boldsymbol{x}^{-})>\log(\alpha_c)$ **faire**
6: $\quad\gamma_j=\frac{\langle\boldsymbol{r}_j,\boldsymbol{r}_j\rangle}{\langle\boldsymbol{d}_j,\boldsymbol{Q}\boldsymbol{d}_j\rangle}$
7: $\quad\boldsymbol{u}_{j+1}=\boldsymbol{u}_j+\gamma_j\boldsymbol{d}_j$
8: $\quad\boldsymbol{r}_{j+1}=\boldsymbol{r}_j-\gamma_j\boldsymbol{Q}\boldsymbol{d}_j$
9: $\quad\beta_j=\frac{\langle\boldsymbol{r}_{j+1},\boldsymbol{r}_{j+1}\rangle}{\langle\boldsymbol{r}_j,\boldsymbol{r}_j\rangle}$
10: $\quad\boldsymbol{d}_{j+1}=\boldsymbol{r}_{j+1}+\beta_j\boldsymbol{d}_j$
11: $\quad j=j+1$
12: **fin tant que**
13: $\alpha=exp(\boldsymbol{r}_j^{\mathrm{t}}(\boldsymbol{u}_j-2\boldsymbol{x}^{-}))$
14: tirer $u\sim\mathcal{U}(0,1)$
15: **si** $u\leq\alpha$ **alors**
16: $\quad\boldsymbol{x}^{+}=-\boldsymbol{x}^{-}+\boldsymbol{u}_j$
17: **sinon**
18: $\quad\boldsymbol{x}^{+}=\boldsymbol{x}^{-}$
19: **fin si**

Moyenne (écart-type)	γ_b	$\gamma_x \times 10^{-4}$	x_i
Cholesky	102,1 (0,56)	6,1 (0,072)	104,6 (9,06)
TPO, $r_{\max} = 10^{-4}$	0,3 (0,06)	$45\ (0,87)$	102,2 (3,30)
TPO, $r_{\max} = 10^{-6}$	6,8 (0,04)	$32\ (0,22)$	104,8 (2,34)
TPO, $r_{\max} = 10^{-8}$	71,7 (0,68)	$21\ (0,29)$	102,7 (2,51)
RJ-PO, $\alpha_c = 0,1$	102,0 (0,54)	$6,1\ (0,081)$	106,1 (9,52)
RJ-PO, $\alpha_c = 0,5$	102,1 (0,55)	$6,1\ (0,074)$	104,2 (8,43)

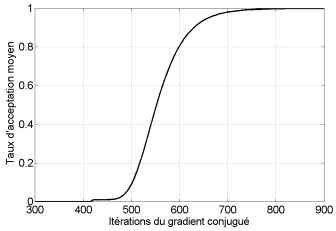

FIGURE 1 – (Gauche) comparaison des méthodes en terme de moyenne et d'écart-type des paramètres estimés ; (Droite) taux d'acceptation moyen en fonction du nombre d'itérations du gradient conjugué.

3 Super-résolution non supervisée

Nous reprenons pour illustration le problème de la super-résolution non supervisée en imagerie, traité dans [5]. Le but est d'obtenir une image nette de résolution supérieure à partir de plusieurs observations floues de basse résolution. Le modèle d'observation de ce problème inverse linéaire est donné par $\boldsymbol{y} = \boldsymbol{M}\boldsymbol{x} + \boldsymbol{b} = \boldsymbol{P}\boldsymbol{H}\boldsymbol{x} + \boldsymbol{b}$, avec $\boldsymbol{y} \in \mathbb{R}^M$ le vecteur des images observées, $\boldsymbol{x} \in \mathbb{R}^N$ l'image nette haute résolution à récupérer, $\boldsymbol{H}$ la matrice $N \times N$ de convolution associée au flou, $\boldsymbol{P}$ la matrice $M \times N$ de décimation et $\boldsymbol{b}$ le bruit additif.

Afin de résoudre ce problème par inférence Bayésienne, le bruit est modélisé par une loi $\mathcal{N}\left(\boldsymbol{0}, \gamma_b^{-1}\boldsymbol{I}\right)$, et l'image $\boldsymbol{x}$ par une distribution $\mathcal{N}\left(\boldsymbol{0}, (\gamma_x \boldsymbol{D}^{\mathrm{t}}\boldsymbol{D})^{-1}\right)$, avec $\boldsymbol{D}$ la matrice de convolution associée au filtre laplacien dont la largeur à mi-hauteur vaut 4 pixels. Les deux hyperparamètres γ_b et γ_x ont quant-à eux un à priori de Jeffrey, non informatif.

Pour échantillonner la distribution à postériori $P(\boldsymbol{x}, \gamma_x, \gamma_b|\boldsymbol{y})$, nous utilisons un échantillonneur de Gibbs qui va successivement tirer des échantillons des lois conditionnelles suivantes :

1. $\gamma_n^{(k)} \sim \mathcal{G}\left(1 + \frac{M}{2}, 2/||\boldsymbol{y} - \boldsymbol{P}\boldsymbol{H}\boldsymbol{x}^{(k-1)}||^2\right)$,
2. $\gamma_x^{(k)} \sim \mathcal{G}\left(1 + \frac{N-1}{2}, 2/||\boldsymbol{D}\boldsymbol{x}^{(k-1)}||^2\right)$
3. $\boldsymbol{x}^{(k)} \sim \mathcal{N}\left(\boldsymbol{\mu}^{(k)}, \left[\boldsymbol{Q}^{(k)}\right]^{-1}\right)$ with

$$\begin{aligned} \boldsymbol{Q}^{(k)} &= \gamma_b^{(k)}\boldsymbol{H}^{\mathrm{t}}\boldsymbol{P}^{\mathrm{t}}\boldsymbol{P}\boldsymbol{H} + \gamma_x^{(k)}\boldsymbol{D}^{\mathrm{t}}\boldsymbol{D} \\ \boldsymbol{Q}^{(k)}\boldsymbol{\mu}^{(k)} &= \gamma_n^{(k)}\boldsymbol{P}^{\mathrm{t}}\boldsymbol{H}^{\mathrm{t}}\boldsymbol{y} \end{aligned}$$

Cette dernière étape est réalisée par une factorisation de Cholesky selon la mise-à-jour (3) comme référence, par l'algorithme *Perturbation Optimization* dans sa version tronquée, et par l'algorithme proposé RJ-PO.

Dans cet exemple, nous observons cinq images de 128×128 pixels ($M = 81920$) et nous recherchons une image de 256×256 pixels ($N = 65536$). 1000 échantillons sont générés avec un temps de chauffe de 100 ; on observe les moyennes et écarts-types des deux hyperparamètres et d'un pixel de l'image reconstruite. L'algorithme de *Perturbation Optimization* tronqué (TPO) utilise classiquement la norme du résidu relatif $r_{\max}$ comme critère d'arrêt du gradient conjugué. L'algorithme proposé (RJ-PO) est quant-à-lui directement contrôlé par le taux d'acceptation cible α_c. Les résultats sont présents dans le tableau 1. Ils montrent clairement le biais engendré par la méthode TPO. De plus la figure 1 montre qu'il est possible d'échantillonner parfaitement en effectuant moins de 700 itérations du gradient conjugué, ce qui est très inférieur à N.

En terme de temps de calcul, sur un Intel Core i7-3770, l'approche par Cholesky prenait en moyenne 20.3s et 6Go de mémoire pour générer un échantillon, contre seulement 12.1s et moins de 200Mo pour le RJ-PO avec $\alpha_c = 0.1$.

4 Conclusion

Nous avons montré sur un exemple concret que l'algorithme PO dans sa version tronqué n'était pas convergent. Nous avons également présenté un nouvel algorithme, basé sur les MCMC à sauts réversibles, qui permet, via une étape d'acceptation-rejet, de garantir la convergence vers la loi cible. De plus l'algorithme est contrôlé directement par un taux d'acceptation cible, ce qui garantie une probabilité non nulle. L'algorithme a montré sont intérêt sur un problème de super-résolution où son impact en terme de temps de calcul et de coût mémoire est bien moindre comparé à une approche de type Cholesky.

Références

[1] H. Rue, « Fast sampling of Gaussian Markov random fields », *J. R. Statist. Soc. B*, vol. 63, n°2, pp. 325–338, 2001.

[2] P. Lalanne, D. Prévost et P. Chavel, « Stochastic artificial retinas : algorithm, optoelectronic circuits, and implementation », *Appl. Opt.*, vol. 40, n°23, pp. 3861–3876, 2001.

[3] D. Geman et C. Yang, « Nonlinear image recovery with half-quadratic regularization », *IEEE Trans. Image Processing*, vol. 4, n°7, pp. 932–946, 1995.

[4] G. Papandreou et A. Yuille, « Gaussian sampling by local perturbations », in *Proc. NIPS*, 2010.

[5] F. Orieux, O. Féron et J. Giovannelli, « Sampling high-dimensional Gaussian distributions for general linear inverse problems », *IEEE Signal Processing Lett.*, vol. 19, n°5, pp. 251, 2012.

[6] X. Tan, J. Li et P. Stoica, « Efficient sparse Bayesian learning via Gibbs sampling », in *Proc. IEEE ICASSP*, 2010, pp. 3634–3637.

[7] R. Waagepetersen et D. Sorensen, « A tutorial on reversible jump MCMC with a view toward applications in qtl-mapping », *Int. Statist. Rev.*, vol. 69, n°1, pp. 49–61, 2001.

[8] P. De Forcrand, « Monte Carlo quasi-heatbath by approximate inversion », *Phys. Rev. E*, vol. 59, n°3, pp. 3698–3701, 1999.